Московский государственный университет
имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики

Отчет по заданию №1

Выполнили:
Данько Артем
Ковалева Василина
Шматенко Дарья

Преподаватель:
Гусева Юлия

Москва
2019

Содержание

Постановка задачи

На вход подается информация о:

- закупках (поставки яблок и карандашей два раза в месяц)
- продажах (лог транзакций, по записи на каждую проданную позицию)
- инвентаре (месячные данные общего количества яблок и карандашей на складе)

Данные доступны в формате CSV. Внутри файла данные отсортированы по дате. Нам необходимо получить следующие данные в CSV-файлах:

1. Состояние склада на каждый день;
2. Месячные данные о количестве сворованного товара;
3. Агрегированные данные об объемах продаж и количестве сворованной продукции по штату и году

Ход решения

Будем работать с каждой тройкой файлов, содержащих информацию для определённого магазина штата.
Сначала создадим таблицу, где для каждой даты будет храниться количество купленных яблок и карандашей в этот день.

1. Чтобы узнать состояние склада на определённый день, надо суммировать все произведенные поставки и вычесть общее количество проданных товаров к текущему дню.
2. Чтобы получить число украденных в каждый месяц товаров, нужно сначала из теоретически высчитанного на предыдущем этапе состояния склада в конце месяца вычесть фактическое состояние склада, известное после инвентаризации. То, что мы получим, будет являться суммарным количеством товара, украденного к концу данного месяца, т. е. будет включать в себя товары, которые были украдены и во все предыдущие месяцы. Значит, чтобы получить количество сворованного товара только для текущего месяца, осталось из полученной величины вычесть суммарное количество сворованного товара, украденного к концу уже предыдущего месяца.
3. Для агрегирования данных об объёмах продаж и количестве сворованной продукции по штату и году необходимо найти сумму годовых продаж и суммарное количество украденных товаров по всем магазинам каждого штата.

Описание программы

Заметим, что имена файлов имеют следующую структуру: штат - номер магазина - вид данных (sell / supply / inventory). Создадим список list_of_start_names_of_files имён, который будет включать только часть штат - номер магазина. Каждый элемент из этого списка будем обрабатывать в функции start(name, global_statistic_frame).

- start(name, global_statistic_frame)
  На основе переданных кратких имён (добавив окончания sell / supply / inventory) сформируем полные имена файлов для соответствующего магазина и с помощью функции read_csv, из библотеки pandas, считаем содержимое этих трёх файлов в датафреймы sell, supply, inventory. В таблицах supply и inventory в колонку, отвечающую за индексацию, запишем значения колонки 'date' (теперь дата является индексом) и с помощью функции drop уберём за ненадобностью колонку 'date'. Также, в этих таблицах строковый формат данных, хранящихся в колонках 'date', с помощью функции to_datetime, переведём в специальный формат даты. Затем датафрейм sell обрабатывается в функции sell_parse(sell). Результатом работы этой функции будет датафрейм, где для каждой даты будет храниться количество проданных яблок и карандашей в этот день для данного магазина. После, в функции daily_inventary(supply, sell) подсчитывается состояние склада на каждый день (di) и в функции month_steal(daily_inv, inventory) вычисляется количество украденного в каждом месяце товара (ms). С помощью функции to_csv сохраняем датафреймы в файлы. Затем получим global_statistic_frame, содержащий агрегированные данные об объёмах продаж и количестве сворованной продукции по штату и году, вызывая функцию update_global_statistic_frame

- sell_parse(sell)
  В столбце sku_num датафрейма sell хранится номер транзакции, из которого можно понять было ли продано яблоко или карандаш. К каждому элементу этого столбца с помощью функции apply применим лямбда-функцию, записывающую вместо номера транзакции число -1 или 1 в зависимости от того, что было продано: -1 если яблоко и 1, если был продан карандаш. Создадим новый столбец num и заполним его значениями столбца scu_num.

  Добъёмся того, чтобы для записи, в которой отражена продажа яблока [карандаша], в столбце scu_num стояла 1 [0], а в столбце num был 0 [1]. Для этого в колонке scu_num, отвечающей за яблоки, обнулим все 1, потому что они сообщают о том, что был продан карандаш, и вместо -1 поставим 1. Аналогично в колонке num, отвечающей за карандаши, обнулим все -1, потому что они сообщают о том, что было продано яблоко, и оставим 1. Все эти операции выполнены через функцию apply, которая применяется к каждой записи столбца. Переименуем колонки в соответствии с их назначением: scu_num в apple и num в pen, используя функцию

rename(columns={"sku_num": "apple" , "num": "pen"}) Осталось сгруппировать записи о покупках по дате, сложив строки в каждой группе, получим число проданных в этот день яблок и карандашей.

- daily_inventary(supply, sell)
  Чтобы узнать состояние склада на каждый день, надо суммировать все произошедшие поставки и вычесть общее количество проданных товаров к текущему дню. Для этого создадим с помощью функции DataFrame (columns=['apple, 'pen'], index = sell.index) датафрейм daily_inv с колонками apple и pen и столбцом индексов, как в датафрейме sell, т. е. индексы - даты. В колонки apple и pen запишем с обратным знаком число проданных товаров и используя concat присоединим таблицу с поставками. Сгруппируем записи по дате и сложив строки в каждой группе, получим изменения в состоянии склада (используя groupby.(['date']).sum). Чтобы получить именно состояние склада, надо сложить все изменения состояния склада, накопившиеся к текущему дню. Это выполняет функция cumsum().

- month_steal(daily_inv, inventory)
  Сформируем группы, объединяющие записи о ежедневном состоянии склада по месяцам, с помощью resample('M') и из каждой группы оставим только последнюю по времени запись, используя last(). Вычтя из этой записи, отражающей теоретическое состояние склада, фактическое состояние склада на этот месяц, известное из датафрейма inventory, получим суммарное количество товара, украденного к концу данного месяца. Чтобы найти количество товара, сворованного в течение данного месяца, осталось из текущей величины вычесть значение для предыдущего месяца. Т. е. ту же самую колонку, но все строки которой сдвинуты вниз на единицу (функция shift(1) применённая к колонкам 'apple' и 'pen'). Образовавшийся после сдвига NaN в первой строке заменяем нулём (fillna(0)).

- update_global_statistic_frame(state, sell, steal, global_statistic_frame)
  Сформируем группы, объединяющие записи о ежедневных продажах по годам, с помощью resample('Y') и для каждой группы выполним суммирование. Используя функцию rename, изменяем имена колонок apple и pen. Затем создаём новую колонку 'year', заполнив её датами, хранящимимся в 'index' и оставим в них только года с помощью to_pydatetime().year. Для данных об украденных товарах проведём те же самые действия. Объединим полученные таблицы, сгруппировав по году и просуммировав значения в каждой группе. Так как в созданной таблице m_year_res после группировки, столбец 'year' стал индексом, мы дублируем его значения во вновь созданную колонку 'year'. Затем обновляем итоговую таблицу global_statistic_frame, присоединив m_year_res и сложив значения по группам.

Итоги

Приведём примеры полученных таблиц:

date	*apple*	*pen*
2006-01-01	33271	2574
2006-01-02	31409	2431
2006-03-31	29529	2260
2006-04-30	27732	2107
2006-05-31	25790	1974

Таблица 1: Состояние склада на каждый день

date	*apple*	*pen*
2006-01-31	10.0	11.0
2006-02-28	6.0	6.0
2006-03-31	7.0	6.0
2006-04-30	6.0	14.0
2006-05-31	8.0	1.0

Таблица 2: Месячные данные о количестве сворованного товара

year	state	apple_sold	apple_stolen	pen_sold	pen_stolen
2006	MS	2152006.0	418.0	155633.0	461.0
2007	MS	2150384.0	377.0	154730.0	346.0
2008	MS	2163559.0	383.0	154597.0	382.0
2009	MS	2152502.0	433.0	155409.0	454.0
2010	MS	2149787.0	418.0	155523.0	441.0

Таблица 3: Агрегированные данные об объемах продаж и количестве сворованной продукции по штату и году

Необходимые компоненты

- Библиотеки и функции
  - pandas – библиотека, предназначенная для хранения таблиц. Также содержит огромное количество универсальных функций для их комфортной обрабоки.
  - read_csv(filepath_or_buffer, sep=', ') – считывает csv файл в dataframe, sep – разделитель
  - to_datetime – переводит строку в тип datetime
  - apply(func) – для каждой строки или столбца применяет указанную функцию
  - to_pydatetime().year – разбивает тип datetime на составляющие
  - drop – удаляет указанную строку или столбец
  - concat(objs) – объединяет последовательность таблиц вдоль какой-либо из осей
  - append(other, ignore_index=False, sort=False) – объединяет таблицы
    other – присоединяемая таблица
    ignore_index – если true, не использует колонку индексов
  - groupby.(['date']) – группирует элементы таблицы по колонке date
  - cumsum() – суммирует все предыдущие и текущий элементы
  - resample() – изменяет частоту для datetable по определенному способу
  - shift(1) – сдвигает столбец или строку по какому-либо направлению
  - fillna(0) – заменяет NaN на 0
- Программы
  - Jupyter Notebook

Участники

Данько Артем, Ковалева Василина, Шматенко Дарья.
Все этапы работы были выполнены совместно.